Remarks

Applicants respectfully request reconsideration of the present application in view of the foregoing amendments and the following remarks. Claims 1, 3-5, and 7-33 are pending in the application. Claims 1, 3-5, and 7-33 are rejected. No claims have been allowed. Claims 1, 19, and 33 are independent.

Cited Art

The Action cites Davidson, U.S. Patent No. 6,083,276 (hereinafter "the Davidson patent").

Claim Rejections under 35 U.S.C. § 112

The Action rejects claims 1, 3-5, and 7-33 under 35 U.S.C. § 112, first paragraph, as failing to comply with the written description requirement. Specifically, the Action alleges that:

> the language recited as 'receiving a reflection of the executable . . . program during a first execution of a program' in conjunction with 'during the second execution of the . . . program' is not shown to have proper support or clear description in the Specifications. Scanning the term 'reflection', it is observed that 'reflection' is but merely metadata regarding types or data structures used in a program (Specifications: page 10 and Fig. 5) such that *the program* in that context is not an executable being run (as recited in the claim) in the course of which one first runtime metadata is collected. . . . This 'first execution' for enabling a concurrent reception of 'reflection' data will not be given any patentable weight . . . .

[Action, at § 4, page 3; emphasis in original.] The Action also treats various "second execution" language as failing to comply with § 112 as well:

> The 'testing during the/a second execution' (cl. 1, 19, 33) and 'during a subsequent second execution' (cl. 1) limitations depend upon the 'during a first execution' limitation, hence would also be treated as lacking proper support.

[Action, at §4, page 3.] Applicants respectfully disagree with these rejections.

While the Action cites to page 10 and states that program reflection is not performed during execution of the program, the passage beginning on page 10 does indeed describe *reflection of program information during execution:*

> At 510, the domain configuration manager 410 reads a reflection related to the computer program whose data domain is being configured. The reflection of a

> computer program in part comprises the meta data related to the data structure elements used in the program. For example, by reading a program reflection the domain configuration manager may be able obtain data structure information such as the description of all the various data types, fields, methods and their parameters within a computer program. *Developer tools currently available provide for features whereby the reflection of a computer programs can be read* to obtain the information related to the data structure elements employed in the programs. For example, *.NET framework by Microsoft® Corporation supports features for allowing an executing program to introspect upon itself to obtain information (meta data) related to its classes or data types, fields, methods, parameters etc. The Java® programming language by Sun® Microsystems allows for similar features.* Older programming languages such C, FORTRAN, and even C++ do not have such features. However, *some such languages are supported by the .NET® framework* (e.g., C, C++, FORTRAN etc.) and *once compiled within the context of the .NET® framework these same reflection features may be used to obtain the reflection of programs coded in all the .NET® supported languages.* Methods other than those described with respect to Java® and the .NET framework may be used as well.

[Application, at page 10, line 15 to page 11, line 4; emphasis added.] Applicants make two observations about this passage.

The first is that the passage indeed describes reflection as obtaining program information *during execution of the program.* This is shown directly by the passage's description of the .NET framework's features for "allowing an executing program to introspect upon itself" to obtain program information, as quoted above. [Application, at page 10, lines 25-26.] This is also shown in the subsequent discussion of the .NET framework, which shows that programs which are *compiled within this framework* feature reflection features. [Application, at page 11, lines 1-3.] Applicants note that, as the Application mentions the reflection is performed *after compilation,* it is performed on the executable program, and in order to use the framework features incorporated in to the program during compilation, it is performed during execution of the program. Another example of using a reflection of a *compiled* program to access program information, as well as a motivation for using reflection versus program information obtained via other means, can be found at page 13:

> For example, *if a program has been compiled using the tools of the .NET® framework, upon reading a reflection of such a program the domain configuration manager 410 should have access to all the methods of the .NET® framework libraries and these methods can be used* in the expression for configuring the data domain of the program for its testing or verification. *The*

> *same may be applicable to the reflection of a Java® program. Furthermore, through the reflection all the methods of all of the data types of a program may be made available to be used in an expression for configuring the domain of any data structure element of the program regardless of the visibility rules applicable to the various methods.* For example, in a program if a data type is defined with a class declaration including several methods that have their visibility rules set to "Private", normally such methods would not be accessible outside of the class declaration itself.

[Application, at page 13, lines 12-26; emphasis added.]

Second, Applicants note that, despite the Action's argument that "'reflection' is but merely metadata regarding types or data structures used in a program," that the above-quoted passage shows that just because a program offers metadata in a reflection operation, this does not mean this reflection is not performed during execution. Indeed, as the passage describes, the ".NET framework . . . supports features for allowing an executing program to introspect upon itself to obtain information (meta data) related to its classes or data types, fields, methods, parameters etc." This example shows that the offering of metadata by a program is not inconsistent with reflection during execution.

Applicants also note that while these particular features are described with respect to the .NET framework, at least one alternative implementation for reflection during execution is described in the Application, namely reflection using features of the Java programming language. As such, Applicants do not wish to imply that said reflection is only possible using the .NET framework, or any other particular framework.

For at least these reasons, Applicants respectfully argue that the specification does indeed describe the "reflection of the executable computer program during a first execution" language of claim 1 and the "reflection of the executable computer program during a first execution of the program" language of claims 19 and 33 in such a way as to comply with the written description requirement of 35 U.S.C. § 112. Additionally, because the claims satisfy the requirements of § 112 with respect to this "first execution" language, Applicants also respectfully note that the "second execution" language of independent claims 1, 19, and 33 are in compliance with § 112 as well.

Applicants respectfully request that the rejection under § 112 of independent claims 1, 19, and 33, as well as that of dependent claims 3-5, 7-18, and 20-32, be withdrawn. Applicants also respectfully request that the claim language be accorded proper patentable weight. Applicants will proceed with patentability arguments which assume the language is given proper patentable weight.

Claim Rejections under 35 U.S.C. § 103(a)

The Action rejects claims 1, 3-5, and 7-33 under 35 U.S.C § 103(a) as unpatentable over the Davidson patent. Applicants respectfully submit the claims in their present form are allowable over the cited art. The cited art does not teach or suggest at least one limitation of each of these claims. Accordingly, Applicants request that all rejections be withdrawn. Claims 1, 19, and 33 are independent.

Claim 1

Claim 1, as amended, recites, in part:

> *reading a reflection of the executable computer program during a first execution of the program;*
>
> producing the data domain based on the domain configuration information and the reflection of the executable computer program, *the data domain representing a limited set of data values to be used as input during a subsequent second execution* of the computer program for testing the executable computer program when executed;
>
> *targeting testing during the second execution* of the executable computer program to use only values for the data structure element that fall within the data domain; and
>
> *determining whether the executable computer program behaves correctly during the second execution* when executing using targeted values falling within the data domain as input.

[Emphasis added.] Support for the above-emphasized language can be found, for example, as the Action notes (at §4, page 3 of the Action) at pages 10 and 18 of the Application.

The Davidson patent does not teach or suggest "reading a reflection of [an] executable computer program during a first execution," "targeting testing during the second execution of the executable computer program" or "determining whether the executable computer program behaves correctly during the second execution when executing using targeted values" because the Davidson patent's actions are performed in order to generate a program, and therefore are *done on an incomplete program.* In its rejection of claim 1, the Action acknowledges that these elements of the claim are not described in the Davidson patent:

> []Davidson does not explicitly teach an executable program to target efficient testing the program, the program executed in compiled executable form.

[Action, at § 6, page 6.]

The Action also argues, as discussed above, that, because the Action had determined the claims as lacking written description under 35 U.S.C. § 112, that the various "first execution" and "second execution" language did not hold patentable weight. [*Id.*] As discussed above, Applicants believe that the language, is well supported in the specification and has patentable weight and should be afforded such weight during subsequent examination. Applicants respectfully request that the claim language be re-considered relative to the Davidson patent in light of the discussion above. In particular, Applicants note that, for reasons discussed at great length in the previous two amendments, the Davidson patent does not teach or suggest "targeting testing during the second execution of the executable computer program" after "reading a reflection of [an] executable computer program during a first execution" as recited in claim 1.

The Action also argues that it would have been obvious to modify the Davidson patent to "implement Sun Microsystems API in testing a target bean code as endeavored by Davidson, so that reflection data is obtained using such runtime API . . . and to test such target compiled bean." [Action, at §6, pages 6 and 7.] In doing so, the Action argues that the Davidson patent "discloses an execution environment . . . to dynamically determine . . .some expected value or range of Java elements . . ., such runtime determination being based on . . . domain data," citing to a "Generate Error" step 458 from Figure 4D of the Davidson patent. [*See*, Action, at § 6, page 6.]

Applicants respectfully traverse this argument. Step 458, which states "Generate Error" is part of Figure 4D of the Davidson patent. The Davidson patent states very clearly, however, that the process of Figure 4D is still performed to generate a program, and is not performed while the target program is executing:

> *Referring now to FIG. 4D, there is shown a method of processing 406 the components 212 to launch the component-based application 214* in accordance with a preferred embodiment of the present invention.

[Davidson patent, at column 27, lines 1-4; emphasis added.] The Davidson patent clearly spells out that this process is performed *prior to execution* further down in column 27:

> In one embodiment of the invention, *after all of the components 212 have been processed,* the init() and start() methods of the root component 212 are invoked *in order to begin actual execution* of the application 214.

[Davidson patent, at column 27, lines 43-46; emphasis added.] As such, even if, for the sake of Argument, the process of Figure 4D in the Davidson patent were considered to read on some form of "testing," the Davidson patent's processing of components does not teach or suggest, and in fact teaches away from "reading a reflection of [an] executable computer program during a first execution" and "determining whether the executable computer program behaves correctly during the second execution" because the Davidson patent's processing of components is performed *before beginning execution.* In fact, Applicants note that the process of Figure 4D of the Davidson patent is simply an extension of the process of figure 4C, which Applicants previously argued was performed in order to generate a program, and is therefore done on an incomplete program, which is not executable.

For at least these reasons, the Davidson patent does not teach or suggest at least the above-quoted language of claim 1. Claim 1, as well as claims 3-5 and 7-18, which depend from claim 1, are thus allowable, and Applicants request their allowance. Applicants will not belabor the merits of the separate patentability of dependent claims 3-5 and 7-18.

Claim 19

Claim 19 recites:

> *producing a reflection of the executable computer program during a first execution of the program;*
>
> *using the reflection of the executable computer program* to produce the data domain for the data structure element according to the domain configuration information, *the data domain representing a limited set of data values to be used as input for testing the executable computer program when executed*; and
>
> *controlling testing during a second execution of the executable computer program to use only values for the data structure element that fall within the data domain.*

[Emphasis added.] For at least the reasons discussed above with regard to claim 1, the Davidson patent does not teach or suggest the above-quoted language of claim 19. Claim 19, as well as claims 20-32, which depend from claim 19, are thus allowable, and Applicants request their allowance. Applicants will not belabor the merits of the separate patentability of dependent claims 20-32.

Claim 33

Claim 33 recites:

> means for *obtaining, on the computer apparatus, a reflection of the executable computer program during a first execution* of the program;
>
> means for processing, on the computer apparatus, the domain configuration information and the reflection to produce and output the data domains corresponding to the data structure elements, *the data domains representing a limited set of data values to be used as input during a second execution of the computer program for testing* the computer program when executed; and
>
> means for *limiting testing during the second execution of the executable computer program* to use only values for the data structure element that fall within the data domain.

[Emphasis added.] For at least the reasons discussed above with regard to claim 1, the Davidson patent does not teach or suggest the above-quoted language of claim 33. Claim 33 is thus allowable, and Applicants request its allowance.

Interview Request

If the claims are not found by the Examiner to be allowable, the Examiner is requested to call the undersigned attorney to set up an interview to discuss this application.

Conclusion

The claims in their present form should be allowable. Such action is respectfully requested.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 595-5300
Facsimile: (503) 595-5301

By ______________________
Stephen A. Wight
Registration No. 37,759